**REMARKS**

In section 3 of the Office Action, the Examiner rejected claim 5 under 35 U.S.C. §112, first paragraph as failing to comply with the written description requirement. Claim 5 recites that the method of claim 1 is performed without identifying the content recipient to the content provider. The Examiner apparently is of the opinion that a user cannot visit a web site without the user identifying himself or herself to the web site.

However, as the Examiner will appreciate, when a user visits a web site, a port is opened on the user's computer to that web site so that two way communications between the user and the web site can be conducted. The web site does not require a user identification nor does the user provide an identification. Thus, the user is not required to provide the user's name, e-mail address, post office address, telephone number, or any other indicia that identifies the user.

When the content recipient visits a web site for the first time, the web site typically stores a cookie on the content recipient's computer. Thus, when the content recipient subsequently visits the web site, the web site reads the cookie to determine that it has previously been in communication with the computer of the

content recipient. A cookie does not identify the content recipient, but merely indicates that the web site has had previous contact with a computer.

A cookie is one way in which the web site can determine at the block 114 of Figure 7 whether the computer of the content recipient has made previous requests for content. However, as disclosed in the present application, this identification need not, and preferably does not, identify the content recipient, and is only sufficient to determine which content, if any, has been previously supplied by the content provider to the requesting content recipient. A cookie meets these requirements because a cookie does not identify the content recipient but only a computer.

Accordingly, claim 5 meets the written description requirement of 35 U.S.C. §112, first paragraph.

In section 5 of the Office Action, the Examiner rejected claims 15 and 16 under 35 U.S.C. §112, second paragraph as being indefinite. The Examiner asserts that applicants fail to provide a distinction between burn through a session and display in front of a session.

Specifically, dependent claim 15 recites that the posted content can burn through a session so that

posted content is visible to a user, and dependent claim 16 recites that posted content can be displayed in front of the session. Thus, dependent claim 16 is the broader of these two claims and recites that the content is displayed in front of a session whether the content is layered over the session in the traditional Windows® vocabulary, or the content is burned through the session as shown in Figure 6 of the present application so that a border 94 is provided around the content.

Accordingly, claims 15 and 16 meet all of the requirements of 35 U.S.C. §112, second paragraph.

In sections 7-15 of the Office Action, the Examiner rejected claims 1-4, 6-13, 17-22, 26, 28-34, 36, 37, and 40-42 under 35 U.S.C. §102(e) as being anticipated by the Chile patent.

Independent claim 1 is directed to a method performed at a content recipient comprising executing first program code at the content recipient so as to identify a content provider having posted content of interest to the content recipient, and executing second program code at the content recipient so as to automatically initiate a request for the posted content.

The Examiner asserts that the Chile patent discloses the invention of independent claim 1 at column 3, line 35 through column 4, line 65.

This portion of the Chile patent discloses a client-based application that automatically updates client-resident software via a network server without user intervention. An associated update script is stored on the network server for each client-resident software. Each script contains a version number to which a corresponding client-resident software will be updated, an indication as to whether the update should proceed using the update script or through a web site, and, for a script-based update, a list of all update files specific to each different operating system which the product supports.

If the update is to occur through the script itself, then the client updating application 500 downloads those update files as specified by the script. If the update is to occur through a web site, the client's web browser is launched by the client updating application 500 and a URL of that web site is passed to the browser. The user then interacts through the browser with the web site to update the client-resident software.

Each client-resident software that seeks an update registers itself during its installation on the client PC.

Once the update script is downloaded, the software updating application 500 determines, based on the version number of the update and that of the client-resident software then being updated, whether the update represents a later version of the software. If not, no update is performed. If it is a later version, then the software updating application examines the remainder of the update script. The script specifies whether the remainder of the update is to governed by the script or is to proceed through a web site.

More specifically, the updating of the client-resident software is performed by the updating application 500 shown in Figures 5A-5D. Upon occurrence of (i) a user-configured date for an update of client-resident software or (ii) an immediate user update request for that client-resident software, the updating application 500 executes.

A block 503 reads a URL for the update site for the client-resident software and establishes a connection to this site. If the connection is not made, flow proceeds to a block 588 which schedules a date for the

next update for this client-resident software. A block 590 then determines whether the user is to confirm updates. If the user requires such confirmation, a block 594 notifies the user of the next scheduled update and requests the user to confirm it. If the user has not requested update confirmation, or after the block 594 notifies the user of the next scheduled update and requests the user to confirm it, a block 596 logs the inability to connect to the update site execution exits from application 500.

On the other hand, if a connection is successfully established, a block 510 constructs a file name for an update script, and issues a download request for this particular file from a script directory associated with the product and residing on the FTP server. A block 511 determines if the download is successful. If the download failed, a block 586 terminates the connection to the site, and blocks 590, 594, and 596 are executed.

If the download is successful, a block 514 reads the name of the client-resident software in a registry 247 and from the downloaded update script. A block 515 determines whether these names match. If the names do not match, flow is routed to the block 586. If

the names match, a block 518 compares the version number associated with the update against that of the client-resident software. If the version numbers match, there is no need to update the installed version. Therefore, a block 519 determines whether an update is available for the client-resident software as currently installed on the client PC. If no update is available, flow is routed to the block 586.

However, if an update is available, a block 522 determines whether the user is to confirm updates. If the user has required such confirmation, a block 525 displays the version number of the update that is available for the installed client-resident software and asks the user whether the update should now proceed. If the user has indicated that the product should not be updated, flow is routed to the block 588. If the user has indicated that the update should now proceed, or if the user requires no confirmation, a block 534 determines whether the update is to proceed through a custom web site or through the update script.

If the update is to proceed through the update script, a block 537 detects the specific O/S operating on the client PC and processes those sections of the script which are O/S-independent and correspond to the detected

O/S. A block 538 downloads each of the copy files specified in the script, and a block 539 determines if the download is successful. If the download is not successful, a block 542 terminates the connection and flow is routed to the block 588. If the download is successful, a block 543 downloads each of the run files and then executes each of these run files in the order downloaded.

A block 545 tests whether all the run files were successfully processed. If not, a block 547 terminates the connection and flow is routed to the block 588. If all run files were successfully processed, a block 549 waits for the last run file to complete its execution and a block 550 automatically schedules a date for the next update of the client-resident software.

A block 552 determines whether the user requires notification of each update and entry of the user's subsequent acknowledgement of its completion. If so, a block 556 prompts the user to confirm that the update has completed. If the user confirms that the update is completed, a block 562 updates the version number in the registry for the client-resident software, and a block 563 determines whether the update script specifies that the user is to re-boot the client PC in

order to complete the update. If such a re-boot is required, a block 566 displays a notification to the user to re-boot the client PC, and a block 567 terminates the connection and flow is routed to a block 582 which determines whether the user is to confirm updates. If the user requires such confirmation, a block 585 notifies the user of the next scheduled update and requests the user to confirm it. Once the user so confirms the update, a block 587 logs the successful script-based update and execution exits.

If at the block 534 the update is to proceed through a custom web site, a block 569 terminates the connection to the update site and launches a browser at the client PC with the URL of the update site. A block 570 determines whether the browser has connected to the update site and opened the page. If the browser is unable to establish a connection to the site and specifically to open the page, flow is routed to the block 582. If the browser is able to open the page at the update web site, the user then interacts, through the client PC and the browser, with the update web site to download the update files for the client-resident software and install the update files accordingly. A block 573 waits until the user has closed the browser.

Once this occurs, a block 574 automatically schedules the next update for the client-resident software, and a block 575 determines, based on questioning the user and soliciting an appropriate response, whether the custom web site update completed.

If the user indicates that the update is completed, a block 577 determines whether the update script specifies that the user is to re-boot the client PC in order to complete the update. If such a re-boot is required, a block 580 displays a suitable notification to the user to re-boot the client PC and a block 581 updates the version number, stored in the O/S registry, for the client-resident software. Execution then proceeds to the block 582.

As can be seen, the Chile patent requires the user either to manually enter a future date upon which a check is to be made to determine whether updates are available, or to manually initiate the update process immediately. Thus, the Chile patent mischaracterizes the update function as without user intervention because, as Figure 5A shows at the block 503, the user in both instances is required to manually initiate the update function. In one instance, the initiation is delayed until the specified date is reached. In the other

instance, there is no delay. Either way, the initiation is manual and not without user intervention.

Accordingly, the Chile patent does not anticipate the invention of independent claim 1.

Independent claim 18 is directed to a computer readable storage medium that stores program code which, when executed by a computing device, automatically initiates a request for the download of content posted by a content provider, and receives the downloaded posted content in response to the request.

Again, the Chile patent requires the user either to manually enter a future date upon which a check is to be made to determine whether updates are available, or to manually initiate the update process immediately. Thus, the Chile patent mischaracterizes the update function as automatic because in both instances the user is required to manually initiate the update function. In one instance, the initiation is delayed until the specified date is reached. In the other instance, there is no delay. Either way, the initiation is manual and not automatic.

Accordingly, the Chile patent does not anticipate the invention of independent claim 18.

Independent claim 32 is directed to a method comprising executing first program code at a content provider so as to post content for access by a content recipient, and executing second program code at the content recipient so as to automatically (i) access the content provider and (ii) initiate receipt by the content recipient of the posted content.

As before, the Chile patent requires the user either to manually enter a future date upon which a check is to be made to determine whether updates are available, or to manually initiate the update process immediately. Thus, the Chile patent mischaracterizes the update function as automatic because in both instances the user is required to manually initiate the update function. In one instance, the initiation is delayed until the specified date is reached. In the other instance, there is no delay. Either way, the initiation is manual and not automatic.

Accordingly, the Chile patent does not anticipate the invention of independent claim 32.

Dependent claims 2, 4, 6, 8, and 33 recite that future requests for the posted content are canceled without communicating such an intent to the content

provider. The Examiner points to column 16, lines 15-40 of the Chile patent for this feature.

This portion of the Chile patent discloses (i) that a block 519 determines whether an update is available, (ii) that, if no update is available, a block 586 terminates the connection to the FTP site, (iii) that if an update is available, a block 522 determines whether the user is to confirm updates, (iv) that if the user has required such confirmation, a block 525 displays the version number of the update and asks the user whether the update should now proceed or not, (v) that a block 528 determines, based on a response from the user, whether the update is to occur now, (vi) that if the user has indicated that the product should not be updated, a block 588 automatically schedules a date for the next update and execution exits, and (vii) that, if the user has indicated that the update should proceed, execution proceeds to a block 534.

As can be seen, the cancellation disclosed in this portion of the Chile patent relates to the current request, not future requests.

Accordingly, the Chile patent does not anticipate the inventions of dependent claims 2, 4, 6, 8, and 33.

Dependent claims 3, 7, 26, 28, and 34 recite that the initiation of the request for the posted content comprises automatic and recurrent initiations of the request for the posted content.

As can be seen from the above description of the Chile patent, the request for downloads is not automatically recurrent. That is, the Chile patent makes clear at column 14, lines 62 and 63 that the dates are user configured so that each time that an update is to be downloaded, the user either sets the download to occur at a specified future date or sets the download to occur immediately. The text of the Chile patent at column 3, lines 40 and 41 does not contradict this description of the Chile disclosure. Accordingly, this updating operation is neither automatic nor recurrent.

Accordingly, the Chile patent does not anticipate the inventions of dependent claims 3, 7, 26, 28, and 34.

Dependent claims 9 and 29 recite that notice is provided to the content recipient when no posted content has been received by the content recipient in response to execution of the second program code.

As can be seen from the flow chart of Figures 5A-5D, no notice is provided even when the download is

not successful as determined by a block 511, no notice is provided even when the product names do not match as determined by a block 515, and no notice is provided even when a download is not available as determined by a block 519. Instead, the connection is merely terminated. The portions of the Chile patent to which the Examiner points do not disclose notice contrary to the explicit rendering of the flow chart.

Accordingly, the Chile patent does not anticipate the inventions of dependent claims 9 and 29.

Dependent claims 12, 20, and 36 recite that the posted content, when received by the content recipient, is displayed behind a session if the session is active.

The Chile patent does not disclose that the updates are displayed, and does not disclose that the updates are displayed behind an active session. Indeed, the client-resident software is updated by the downloaded updates, but there is no disclosure that the updates are displayed. Indeed, displaying the updates would not make sense in the context of the Chile patent.

Accordingly, the Chile patent does not anticipate the inventions of dependent claims 12, 20, and 36.

Dependent claims 17, 30, 41, and 42 recite that the program code at the content recipient which automatically initiates a request for posted content is electronically received at the content recipient.

The Chile patent does not disclose that the software updating application 500 is electronically received at the content recipient.

Accordingly, the Chile patent does not anticipate the inventions of dependent claims 17, 30, 41, and 42.

Dependent claim 40 recites that the program code at the content recipient which automatically initiates a request for posted content is downloaded by the content provider to the content recipient.

The Chile patent does not disclose that the software updating application 500 is downloaded by the web site to the client.

Accordingly, the Chile patent does not anticipate the inventions of dependent claim 40.

In section 17 of the Office Action, the Examiner rejected claims 5, 27, and 35 under 35 U.S.C. §103(a) as being unpatentable over the Chile patent in view of the Kenner patent.

These dependent claims recite that the automatic initiation of requests for posted content are performed without identifying the content recipient to the content provider.

The Examiner recognizes that the Chile patent does not disclose this claimed feature. Therefore, the Examiner points to the Summary and column 10, lines 40-67 of the Kenner patent.

These portions of the Kenner patent disclose that a POST request is used to send user information to the server. The server processes the user information and responds by sending an HTML file. A GET request based on that HTML file is then sent, which results in the codes file being sent. Also, the FTP (File Transfer Protocol) may be used rather than HTTP (HyperText Transport Protocol). Accordingly, the user first sends the command "USER anonymous" to request an anonymous FTP transaction. The server responds by indicating success or failure. The command "PASS jdoe@user.com" is then sent (in anonymous FTP transactions, the user' e-mail address is usually requested as the password). The server responds with success or failure.

Claims 5, 27, and 35 are patentable over the Chile patent in view of the Kenner patent for at least two reasons.

First, whatever the meaning of anonymous as used in the Kenner patent, it does not mean automatic initiation of requests for posted content are performed without identifying the content recipient to the content provider because, as specifically disclosed in the Kenner patent, the user' e-mail address is requested as the password.

Therefore, because neither the Chile patent nor the Kenner patent discloses that automatic initiation of requests for posted content is performed without identifying the content recipient to the content provider, the combination of the Chile patent and the Kenner patent cannot teach the inventions of dependent claims 5, 27, and 35.

Second, the Chile patent discloses that, during registration, the user's identity is provided to the web site providing the update. The user's identity is provided to the web site providing the update typically because the user, if not an authorized user, is not otherwise entitled to updates. The Kenner patent does not disclose how the user can be entitled to updates

without being an authorized user and does not disclose how the user can be determined to be authorized without providing a user identification.

Therefore, because the Kenner patent does not teach how the Chile patent can be modified to provide updates without knowing the identity of the user, the combination of the Chile patent and the Kenner patent cannot teach the inventions of dependent claims 5, 27, and 35.

Accordingly, dependent claims 5, 27, and 35 are patentable over the Chile patent in view of the Kenner patent.

In section 18 of the Office Action, the Examiner rejected claims 14 and 23 under 35 U.S.C. §103(a) as being unpatentable over the Chile patent in view of the Mohammed patent.

These dependent claims recite that notice of the receipt of the posted content is provided to the content recipient in the form of an icon.

The Examiner recognizes that the Chile patent fails to disclose providing notice of the receipt of the posted content to the content recipient in the form of an icon. Therefore, the Examiner points to column 5, lines 25-50 of the Mohammed patent.

This portion of the Mohammed patent discloses that a detection module 154 indicates to the user that an upgrade process is occurring, retrieves a registry entry for the software OS platform, and loads the registry entry. The detection module 154 then compares the parameters OrigPlatform and NewPlatform. If they remain unequal, the software OS platform registry entry has not changed because the upgrade process has failed, and the detection module 154 then exits. If they are equal, the software upgrade was successful and the detection module 154 retrieves the Restart entry from the registry. The Restart entry has three possible values: 0 to force the user to reboot but displaying a message to indicate that the system is about to reboot; 1 to prompt the user to agree or decline with the reboot; and, 3 to perform a hidden reboot in which the user is not notified of the reboot.

As can be seen, there is no mention in the Mohammed patent or the Chile patent of using an icon to indicate that an upgrade has been received and that the computer should be rebooted in order to start using the upgrade.

Therefore, the combination of the Chile patent and the Mohammed patent cannot teach the inventions of dependent claims 14 and 23.

Accordingly, dependent claims 14 and 23 are patentable over the Chile patent in view of the Mohammed patent.

In section 19 of the Office Action, the Examiner rejected claims 15, 16, 24, 25, 38, and 39 under 35 U.S.C. §103(a) as being unpatentable over the Chile patent in view of the Slotznick patent.

Dependent claims 15, 16, 24, 25, 38, and 39 recite that, upon an action related to the notice, the received posted content either burns through a session or is displayed in front of the session so that the posted content is visible to a user.

The Examiner recognizes that the Chile patent does not disclose burning the received posted content through a session or displayed it in front of the session so that the posted content is visible to a user. Therefore, the Examiner points to column 15, lines 1-45 of the Slotznick patent.

This portion of the Slotznick patent discloses that the received secondary information is downloaded into a window which is fully or partially hidden by the

window or frame in which the primary information is being displayed, or the received secondary information is downloaded into a window which is smaller than the information screen to be displayed so that only a portion of the secondary information is displayed, or the received secondary information is downloaded into a window that is not fully displayed. In each case, the window must be activated to bring it to the front.

As can be seen, there is no mention in the Slotznick patent or the Chile patent of burning received posted content through a session or displaying it in front of the session so that the posted content is visible to a user upon an action related to the notice.

Therefore, the combination of the Chile patent and the Kenner patent cannot teach the inventions of dependent claims 15, 16, 24, 25, 38, and 39.

Moreover, the combination of the Chile patent and the Slotznick as suggested by the Examiner would not be obvious because there is no reason to display the downloaded software upgrade.

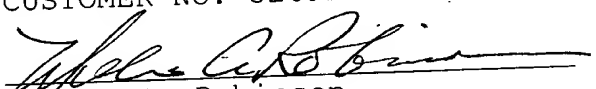
Accordingly, for both of these reasons, dependent claims 15, 16, 24, 25, 38, and 39 are patentable over the Chile patent in view of the Slotznick patent.

**CONCLUSION**

In view of the above, it is clear that the claims of the present application patentably distinguish over the art applied by the Examiner. Accordingly, allowance of these claims and issuance of the above captioned patent application are respectfully requested.

Respectfully submitted,

SCHIFF HARDIN LLP
6600 Sears Tower
233 South Wacker Drive
Chicago, Illinois 60606
(312) 258-5500
CUSTOMER NO. 32692

By: 

Melvin A. Robinson
Registration No.: 31,870
Attorney for Applicants

February 28, 2005

CH1\ 4233006.1